Ex No:	**Managing Tables using DML, DCL and TCL Commands**
Date	25.07.23

Aim:

To execute all commands of data manipulation language to get the desired output.

Description:

DML, DCL, and TCL are important concepts in database management systems (DBMS) that are used to manipulate and control data within a database. Here's a brief overview of each concept:

1. Data Manipulation Language (DML): DML is a set of commands that allows users to retrieve, insert, update, and delete data within a database. The common DML commands include:
   SELECT: Retrieves data from one or more tables based on specified conditions.

   Syntax: select column_1, column_2 from table_name;

   INSERT: Adds new data records into a table.

   Syntax: insert into table_name values (value1, value2);

   UPDATE: Modifies existing data records in a table.

   Syntax: update tablename set columnname = value where condition;

   DELETE: Removes data records from a table.

   Syntax: delete from tablename where condition;

2. Data Control Language (DCL): DCL comprises commands that control access to the database by managing user privileges and permissions. These commands are used to define and manage the security and integrity of the database. Some common DCL commands are:

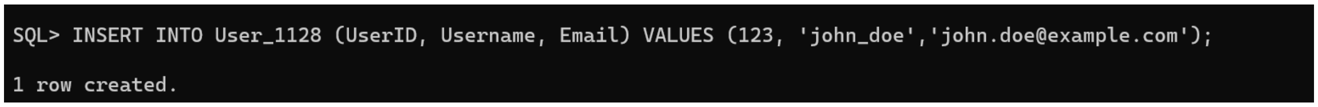
   GRANT: Provides user privileges and permissions, allowing them to perform specific operations on the database objects.
   Syntax: grant insert/ select/ update on table_name to user_name;

   REVOKE: Removes or revokes previously granted privileges from users.
   Syntax: revoke all on table_name from user_name;

3. Transaction Control Language (TCL): TCL is used to manage transactions within a database. A transaction is a logical unit of work that consists of one or more database operations, typically executed as a single unit. TCL commands help maintain the consistency and integrity of the database by controlling the transactional behavior. The main TCL commands are: commit, rollback and savepoint.

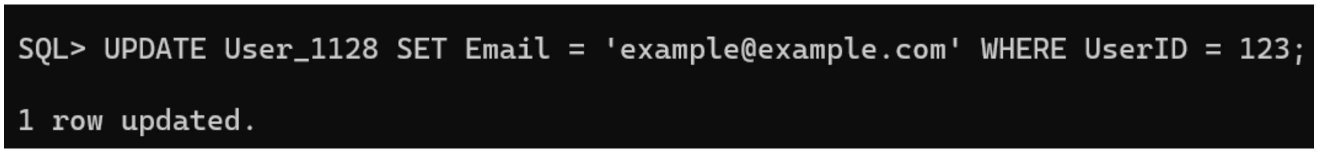
DML (Data Manipulation Language) Questions:

1. Insert a new user into the "User" table.
   Query: insert into user_1128 values(123,'john',john123@example.com','132Q!#',94224234);



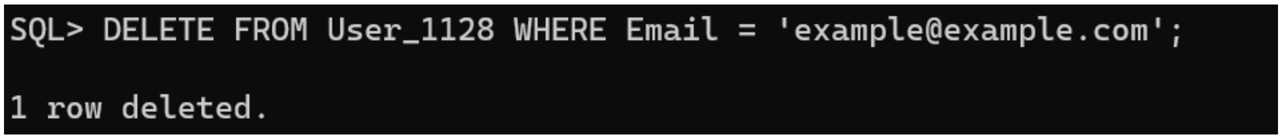
```
SQL> INSERT INTO User_1128 (UserID, Username, Email) VALUES (123, 'john_doe','john.doe@example.com');

1 row created.
```

2. Update the email address of a user with UserID 123.
   Query: update user_1128 set email = 'dbms@gmail.com' where id =123;



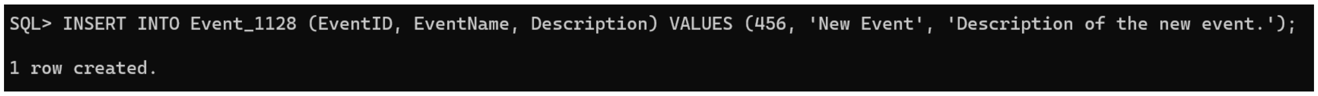
```
SQL> UPDATE User_1128 SET Email = 'example@example.com' WHERE UserID = 123;

1 row updated.
```

3. Delete a user with the email "example@example.com.
   Query: delete from user_1128 where email = 'example@example.com';



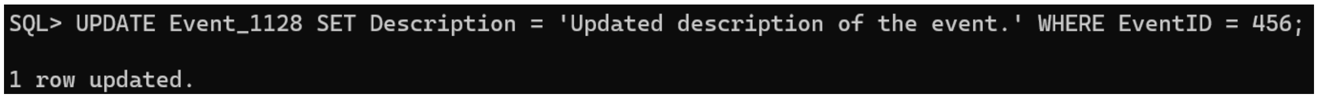
```
SQL> DELETE FROM User_1128 WHERE Email = 'example@example.com';

1 row deleted.
```

4. Insert a new event into the "Event" table.
   Query: insert into event_1128 values(10,'Volley Ball','20', 'Sports');



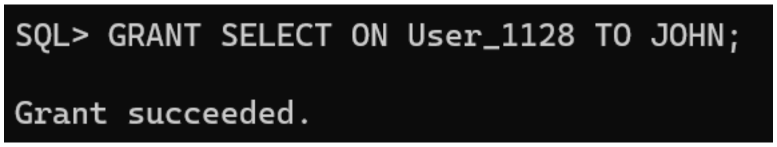
```
SQL> INSERT INTO Event_1128 (EventID, EventName, Description) VALUES (456, 'New Event', 'Description of the new event.');

1 row created.
```

5. Update the description of an event with EventID 456.
   Query: update event_1128 set description = 'ICC' where event_id = 456;



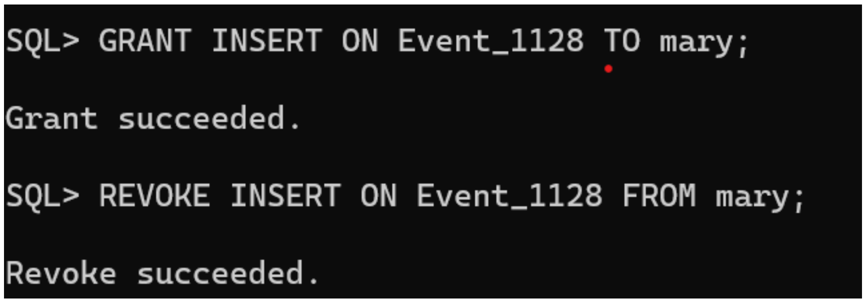
```
SQL> UPDATE Event_1128 SET Description = 'Updated description of the event.' WHERE EventID = 456;

1 row updated.
```

6. Grant SELECT privileges on the "User" table to a user named "john".
   Query: grant select on user_1128 to john;



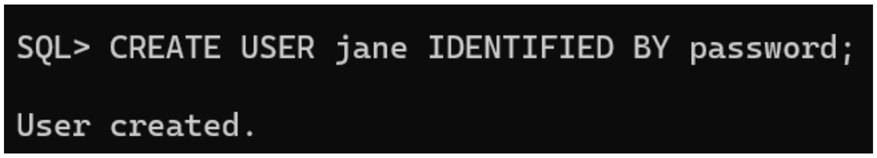
```
SQL> GRANT SELECT ON User_1128 TO JOHN;

Grant succeeded.
```

7. Revoke INSERT privileges on the "Event" table from a user named "mary".
   Query: revoke all on user_1128 from mary;

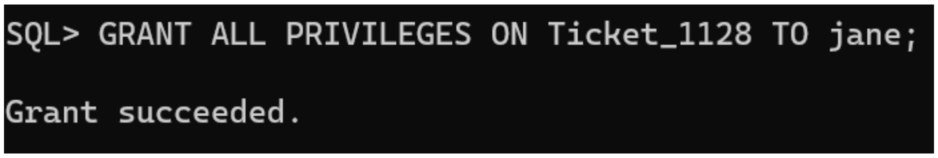


```
SQL> GRANT INSERT ON Event_1128 TO mary;

Grant succeeded.

SQL> REVOKE INSERT ON Event_1128 FROM mary;

Revoke succeeded.
```

8. Create a new user with the username "jane" and grant them all privileges on the "Ticket" table.
   Query: grant all on ticket_1128 to jane;

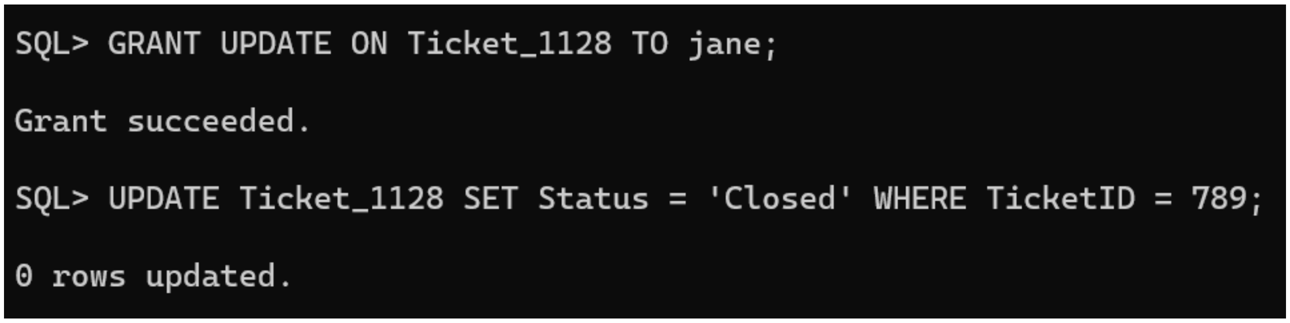


```
SQL> CREATE USER jane IDENTIFIED BY password;

User created.
```



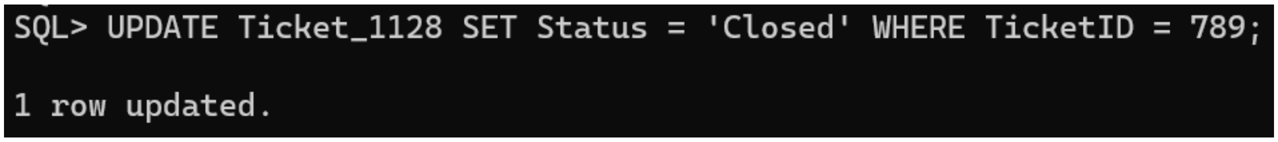
```
SQL> GRANT ALL PRIVILEGES ON Ticket_1128 TO jane;

Grant succeeded.
```

9. Allow the user “Jane” to perform update operation on the “Ticket” table.
   Query: update system.ticket_1128 set seat_number = 25 where ticket_id = 10;



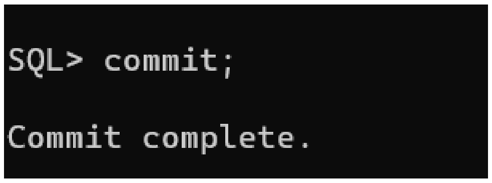
```
SQL> GRANT UPDATE ON Ticket_1128 TO jane;

Grant succeeded.

SQL> UPDATE Ticket_1128 SET Status = 'Closed' WHERE TicketID = 789;

0 rows updated.
```

10. Perform update operation on the “Ticket" table.
   Query: update system.ticket_1128 set Event_ID = 10 where Ticket_ID =112



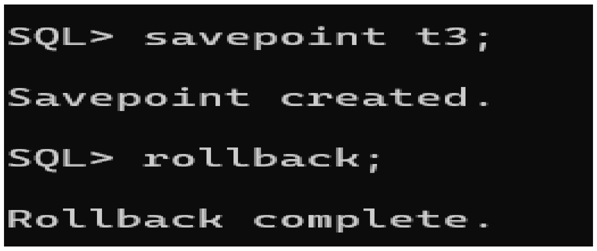
```
SQL> UPDATE Ticket_1128 SET Status = 'Closed' WHERE TicketID = 789;

1 row updated.
```

11. Perform commit a transaction in the database.
    Query: commit;



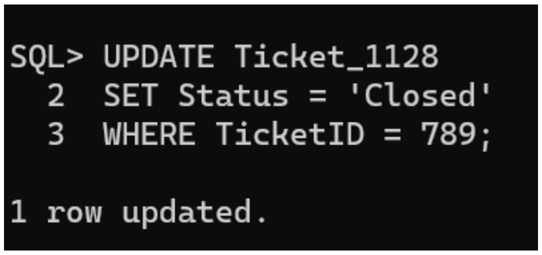
```
SQL> commit;

Commit complete.
```

12. Perform roll back a transaction to a specific savepoint.
    Query: savepoint t3;
    Rollback;



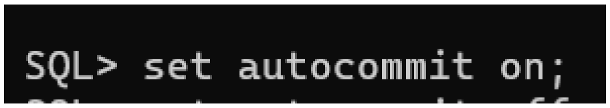
```
SQL> savepoint t3;

Savepoint created.

SQL> rollback;

Rollback complete.
```

13. Perform set a savepoint within a transaction.
    Query: update ticket_1128 set status = ‘Closed’ where ticket_id=10;



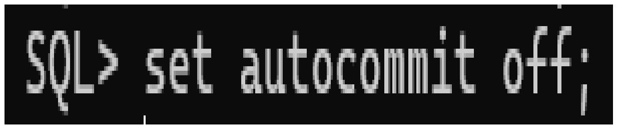
```
SQL> UPDATE Ticket_1128
  2  SET Status = 'Closed'
  3  WHERE TicketID = 789;

1 row updated.
```

14. Enable autocommit mode in the database.
    Query: set autocommit on;



```
SQL> set autocommit on;
```

15. Disable autocommit mode in the database.
    Query: set autocommit off;



```
SQL> set autocommit off;
```

Result:

The DML commands are executed successfully and displayed using sql.